Context and Natural Language in Formal Concept Analysis

Tim Wray and Peter Eklund

IT University of Copenhagen
Rued Langgaards Vej 7 2300 Copenhagen S. Denmark
email: timwray.mail@gmail.com, petw@itu.dk

Abstract. Formal Concept Analysis is an unsupervised clustering technique that accepts as inputs a set of objects with their corresponding attributes (called a formal context) and produces a mathematical lattice containing sets of objects and attributes at each position in the lattice. Each point in the lattice is called a *formal concept* representing the set of objects sharing the same values for a certain set of attributes; each sub-concept in the lattice is a subset of the objects of the concepts above it and a super-set of the attributes above it. This paper concerns generating meaningful natural language fragments from the list of attributes of a formal concept using a predicate map. It therefore address the important problem of determining natural language context in formal concept analysis.

1 Introduction

This paper demonstrates a natural language (NL) generator producing coherent natural language fragments from a formal concept in FCA. It therefore concerns determining natural language context in formal concept analysis.

The paper is structured as follows. Section 2 begins by describing data structures used to store museum metadata as a formal context, the baseline cross-table of objects and attributes that is the input to the FCA algorithm that generates the concept lattice. We illustrate via introducing the data structures and framework methods the problem we subsequently address in Section 3. Section 3 then illustrates the natural language processing pipeline and toolkits that are used to extract attributes from museum metadata and express formal concepts in natural language. In Section 4 we explain how pathways of formal concepts can be connected with natural language, particularly how this problem can be solved for creative works.

2 Preliminaries: FCA and Data structures

The basics of Formal Concept Analysis (FCA) are found in [6, 1]. A formal context is a triple $K := (G, M, I)$ where G describes the set of objects, M, the set of attributes and I the incidence relations between $\langle g, m \rangle \in I$ meaning that object $g \in G$ has attribute $m \in M$. Within our framework (called CollectionWeb),


```
{  "id": "157697",
   "name": "82.65.413",
   "uri": "http://www.brooklynmuseum.org/opencollection/ ... ",
   "title": "[Untitled] (Mother with Children, New York)",
   "objectDate": "1922-1924",
   "objectDateBegin": "1922",
   "objectDateEnd": "1924",
   "medium": "Gelatin silver photograph (from glass plate negative)",
   "description": "Portrait-group Condition: good",
   ...
   "attributeIDs": [8, 10, 107, 333, 974, 2808, ... ]
}
```


Fig. 1. *Untitled (Mother with Children, New York)* by American photographer Consuelo Kanaga, the image associated with a single *BrooklynMuseumObject* record and its (partial) JSON representation. Found at https://www.brooklynmuseum.org/opencollection/objects/157697/.

each object g is represented in as an instance of a *Object*, and each attribute m is represented as an instance of *Attribute*. The formal context, or cross-table of objects and attributes, is stored as a table that references the incidence relations between *Objects* and *Attributes*.

An *Object* instance only records a unique identifier that corresponds to an ID in a formal context along with the museum's identifier for that object. An *Object* can be extended to incorporate additional metadata fields. For instance, in the applications *A Place for Art* [8] and *The Brooklyn Museum Canvas* [7], we store museum metadata using instances of *APlaceForArtObject* and a *BrooklynMuseumObject* – both of which inherit from *Object*. It is possible to create many descendant classes of *Objects* in a single application, allowing a single app to display and process metadata from multiple sources. Fig. 1. is a (partial) JSON representation of a single *BrooklynMuseumObject* showing the metadata associated with a work by photographer Consuelo Kanaga.

There are the two identifier fields associated with the *BrooklynMuseumObject*: *name* which is a unique identifier the museum uses to identify the object, and the *id*, an internal identifier to reference the object within its formal context. *Attributes*, like *Objects* have a unique identifier that corresponds to a formal attribute within a formal context along with a *value* property. The following is a list of attribute identifiers from the above example object record: *[8, 10, 107, 333, 40349, 40344, 974, 2808, 2845, 5643, 34856, 40348]*. Dereferencing these attribute identifiers yields the following attribute values: *[20th century, American, child, 1920s, mother and child, melancholy, children, photograph, New York, mother, Consuelo Kanaga, poverty]*. This description can be formed by constructing a *FormalConcept* object based on its *Object* and *Attribute* identifiers:

```
// Usage: new FormalConcept(obj[], attr[]);
formalConcept =
```

```
new FormalConcept([157697], [8, 10, 107, 333, 40349,
    40344, 974, 2808, 2845, 5643, 34856, 40348]);
```

calling the *getNaturalLanguageExpression()* method yields:

```
An object associated with 20th Century, American children,
    1920s, photograph, New York, mother, Consuelo Kanaga,
    melancholy, poverty,  mother and child
```

By loading the collection's *FormalContext*, and then instantiating that same formal concept to create a *FormalConceptInContext*, we can compute its upper and lower neighbours:

```
// Loads the entire formal context of the collection
K = FormalContext::load();

// Assign the object ID and its attributes
A = [157697];
B = [8, 10, 107, 333, 40349, 40344,
        974, 2808, 2845, 5643, 34856, 40348];

// Creates a new formal concept
formalConcept =  new FormalConceptInContext(K, A, B);

// Retrieves its upper neighbours
upperNeighbours = formalConcept->getUpperNeighbours();

// Display its upper neighbours
for (upperNeighbour in upperNeighbours) {
   getNaturalLanguageExpression(upperNeighbour);}
```

The above "code"[1] outputs the following:

```
objects associated with 20th century,1920s,photograph,mother

objects associated with 20th century, American, child, 1920s,
    children, photograph, Consuelo Kanaga

objects associated with 20th century, American, child,
    children, 1920s, photograph, New York, Consuelo Kanaga,
    melancholy, poverty

objects associated with 20th century, American, child, 1920s,
    photograph, New York, Consuelo Kanaga, melancholy, poverty

objects associated with 20th century, American, child,
    children, photograph, Consuelo Kanaga, mother and child
```

[1] This is not code, rather a conceptual shorthand that abbreviates the presentation.

These NL fragments list only the attribute values of the formal concept. The lists offer little expression of how the objects are grouped and related. Our purpose therefore is to develop meaningful natural language fragments of formal concepts. This is addressed in the next section.

3 Natural language Predicates and Parsers

The framework CollectionWeb uses *Predicates* to add an additional semantic layer to attributes and *Parsers* to express semantics in NL. A *Predicate* is a qualifier that characterises the relationship between the *Attribute* and *Object*. For example, the work in Fig. 1. is *byArtist Consuelo Kanaga* is *typeOf photograph* that *depicts:location New York*. The following is a list of all attributes of the *Untitled* work by Consuelo Kanaga from Fig. 1., supplemented with *Predicates*:

```
fromTimePeriod:20th century
artistNationality:American
depicts:person-subject:child
fromTimePeriod:1920s
depicts:person-subject:mother and child
depicts:emotional-subject:melancholy
depicts:person-subject:children
isTypeOf:photograph
depicts:location:New York
depicts:person-subject:mother
byArtist:Consuelo Kanaga
depicts:emotional-subject:poverty
```

This is identical to the *Subject-Predicate-Object* form in RDF. RDF-like predicates are used to generate NL as follows. *Predicates* exist independently of *Attributes*: an *Attribute* may be assigned to one *Predicate* but a *Predicate* may be assigned to multiple *Attributes*. CollectionWeb does not map explicit relationships between *Predicates*, nor does it allow more than one *Predicate* to be assigned to a single *Attribute* instance. In such cases, if a single *Attribute* is assigned to the same *Object* but with two different *Predicates* – e.g. an *Object* maybe *associatedWithAgent:Consuelo Kanaga* and/or created *byArtist:Consuelo Kanaga*. CollectionWeb treats both occurrences as two separate *Attributes*, each being potentially assignable to the object. *Predicates* can be used to determine how an *Attribute*, or set of *Attributes*, share a common root *Predicate*. For example, the following attributes:

```
fromTimePeriod:20th century
fromTimePeriod:1920s
fromTimePeriod:1930s
fromTimePeriod:1940s
```

If assigned as *Attributes* to a *FormalConcept*, the following results from the *this.getNaturalLanguageExpression()* method: “works from the 1920s to the

1940s". This is because the attributes are assigned to the *fromTimePeriod* predicate, a built-in predicate used to express common metadata fields found within museum collections. Each built-in *Predicate* is linked to a *Parser*.

The *Predicate* describes the semantic relationship between an object and an attribute and contains simple NL processing rules and templates. A *Parser* is *procedural* information about how a class of attributes should be extracted from free-text. In this example, the *fromTimePeriod* predicate uses the *TimePeriod-Parser* to express *fromTimePeriod* attributes as an individual decade, a date range or a century. Likewise, the parser can also be used to extract dates from free text metadata from a object that expresses a date or date range.

The CollectionWeb framework contains several built-in *Predicates* – each with their own *Parser* – that allow NL extraction and expression of common metadata found in museum collections – date ranges, artwork series, mediums, keywords and titles. The parsers use the Illinois Part of Speech Tagger[2] [5] to recognise nouns and noun phrases, and the Stanford Named-Entity Recogniser[3] [3] to identify people, places and organisations. Other *Predicates* built into the CollectionWeb framework are as follows:

(1) **isTypeOf** uses the *ObjectNounParser* to extract and display text that describes object types. For example, attribute values that represent category types and classes such as *Contemporary works* or *paintings* use the *isTypeOf* predicate. The *ObjectNounParser* ensures that object nouns are represented in the correct form when expressed in singular or plural. When parsed from text, the parser ensures that object nouns are represented as collective-nouns (e.g. *warfare objects* instead of just *warfare*) and that object nouns are expressed in singular and plural form. As an example of how the *isTypeOf* predicate expresses attributes, consider the following:

```
isTypeOf:photograph
isTypeOf:drawing
isTypeOf:painting
```

If assigned as *Attributes* to a *FormalConcept*, the *this. getNaturalLanguage-Expression()* method: produces "paintings, drawings and photographs".

(2) The **is** predicate uses the *ObjectDescriptionParser* to parse descriptive elements and non-iconographical concepts from textual data that describe a literal depiction of an object by extracting noun-phrases. It also implements a custom method for expressing those entities in a meaningful way.

ObjectDescriptionParser works well if the artwork's title is the data source, although short snippets describing the work can also be used. The parser itself does not distinguish the iconography in the source text, it assumes all text is a literal description of the work, rather than iconography. As an example of how the *is* predicate expresses attributes, the following attributes:

```
is:cover
is:box
```

[2] http://cogcomp.cs.illinois.edu/page/download_view/POS
[3] http://nlp.stanford.edu/software/CRF-NER.shtml

If assigned as *Attributes* to a *FormalConcept*, the *this. getNaturalLanguage-Expression()* method produces: "objects that consist of boxes and covers"

(3) The **Depicts** predicate uses the *ImageSubjectParser* to parse entities and iconographical concepts from snippets describing image-based artwork by extracting noun-phrases and named-entities. The parser recognises the difference between iconographical concepts representing objects, people or places, and implements custom methods for expressing entities.

Although *ImageSubjectParser* can accept any data, the parser is designed to work on artwork titles for two-dimensional visual works – paintings, prints, photographs or other 'flat' visual media – where the title evokes a depiction, rather than stating what the work is. Depending on how the artist titles a work, the parser could accept titles of works in other mediums – sculptures or 3-dimensional works, e.g., consider how *depicts* expresses the following:

```
depicts:person-subject:child
depicts:person-subject:children
depicts:location:New York
depicts:emotional-subject:poverty
```

If assigned as *Attributes* to a *FormalConcept*, *this. getNaturalLanguageExpression()* produces: "works that depict children in poverty in New York".

CollectionWeb also contains other general-purpose *Parsers* not tied to any specific *Predicate*. (1) The *StopWordsRemovalParser* can be used to tokenise a string removing punctuation, numbers and stopwords. (2) The *NounPhraseParser* extracts head nouns and noun phrases from free-text and is useful for parsing text that depicts a work's medium or title. For instance, the free text – *Woodblock color print with silver pigment* – parses as attributes with values {*'woodblock color print'*, *'silver pigment'*, *'print'*, *'pigment'*}. Likewise, the free text that describes a work's title as "Sculptor's Model of Male Body Wearing Long Skirt" returns attribute values {*'sculptor's model'*, *'male body'*, *'long skirt'*, *'model'*, *'body'*, *'skirt'*}.

Predicates and *Parsers* combined can be used to build complete NL descriptions of any formal concept. CollectionWeb has the ability to specify and configure how the NL phrasing is expressed via configurable JSON files. These files store what CollectionWeb refers to as a *predicate map*, which is a two-dimensional array of *Predicates* where each predicate describes a NL phrase fragment, such as *"made with steel and wood"* or *"depicts farmhouses and animals"*. The predicate map can be used to:

- map associations between NL phrase fragments and *Predicates* that in turn map to the metadata fields within the museum collection.
- describe the order that phrase fragments occur, and prioritize which fragments should be displayed at a given position within the phrase.
- describe, in a declarative manner, how each phrase fragment is displayed for a given set of attribute values.

Fig. 2 shows how a predicate map generates the natural language phrase *"photographs by Consuelo Kanaga from the 1920s made with gelatin silver that*

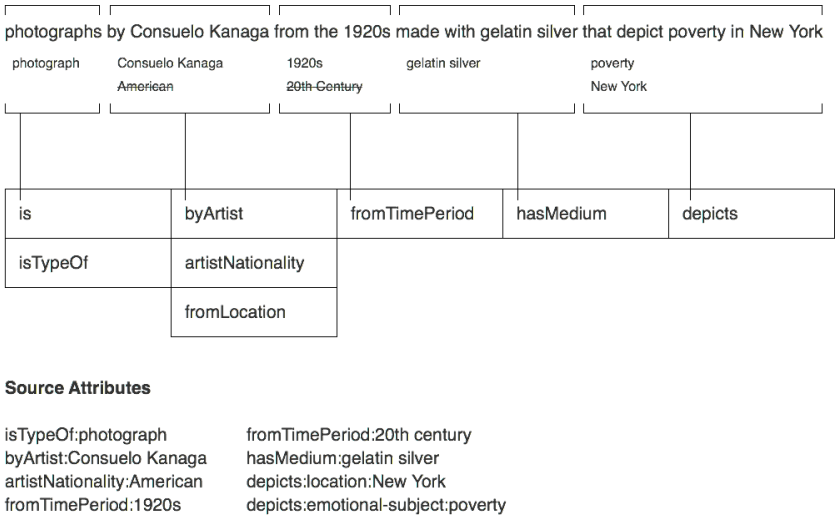

Fig. 2. An example of how a formal concept can be expressed in natural language from its source attributes using a predicate map.

depict poverty in New York" from the attributes of its formal concept. The predicate map is in the centre of the figure, the horizontal ordering of columns corresponds to the ordering of the phrase fragments in the NL statement, and the ordering of each row to the priority of predicates displayed, e.g. as the first column shows *is* and *isTypeOf*, CollectionWeb first searches for attributes in the formal concept with the predicate *is*, if it finds none, it searches for attributes with the predicate *isTypeOf*. The algorithm produces a phrase fragment for the attribute *isTypeOf:photograph* using the *isTypeOf* predicate, which then forms the first part of the phrase.

The algorithm then moves to the second column, searches for attributes with the *byArtist* predicate. This attribute is present – *byArtist:Consuelo Kanaga* – so it renders its phrase fragment and ignores any attributes with the *artistNationality* or *fromLocation* predicates. This ordering can be used to denote an implicit hierarchy of concepts expressed in NL without producing excessively long NL descriptions, e.g., if the attribute *byArtist:Consuelo Kanaga* and *depicts:emotional-subject:poverty* are 'removed' from the formal concept by navigating to its upper neighbour, the generated NL statement would read *"photographs by American artists from the 1920s made with gelatin silver that depict New York"*.

In the column to follow, another attribute is omitted – *fromTimePeriod: 20th century* – but this time, for a different reason. The *fromTimePeriod* predicate uses the *TimePeriodParser*, that omits redundant attributes to describe a century, such as *20th century* when more specific date attributes are present. After the column is processed, the algorithm continues left-to-right within the predicate map, forming the complete NL statement of the formal concept.

As mentioned, the predicates *is*, *isTypeOf*, *fromTimePeriod* and *depicts* are built into the framework, each with a *Parser* dictating how a phrase fragment

is expressed for a given set of attribute values. However, the predicates *byArtist*, *artistNationality*, *fromLocation* and *hasMedium* are not built-in – instead, they are provided in this example as custom predicates. Custom predicates are defined to express any metadata field on an object within a museum collection. They use an additional field to dictate how that predicate is displayed as a phrase fragment, pre-pending any prepositions, suffixes or prefixes so that its attribute values are inserted into the phrase. For example, the custom predicate *hasMedium* uses *made with* `{`*attribute*`}` so that the attribute *hasMaterial:gelatin silver* appears as *"made with gelatin silver"*. Likewise, the custom predicate *artistNationality* uses *by* `{`*attribute*`}` *artists* so that attribute *artistNationality:American* appears as *by American artists*.

Using *Predicates* and *Parsers* – including those built into the framework – the following example shows the list of neighbouring concepts shown previously, at the end of Section 2, but with the natural language rules applied:

> *A photograph by Consuelo Kanaga from the 1920s that depicts children in poverty and melancholy in New York*
> *other photographs from the 1920s that depict mothers*
> *other photographs by Consuelo Kanaga from the 1920s that depict children*
> *other photographs by Consuelo Kanaga from the 20th century that depict mothers and children in poverty and melancholy*
> *other photographs by Consuelo Kanaga from the 1920s that depict children in poverty and melancholy in New York*
> *...*

The list more expressive than those shown at the end of in Section 2. In this case, the upper neighbours of the formal concept represented by the *Untitled* work in Fig. 1. link to Kanaga's other works that depict mothers, children and poverty from a similar time period.

4 Connecting Formal Concepts via Natural Language

Formal concepts can be related in terms of their upper or lower neighbours within the concept lattice or as having a certain degree of *concept similarity* [4] to other concepts within the lattice. In addition to describing concepts individually, CollectionWeb expresses *relations* between two formal concepts in natural language (NL). Expressing the conceptual relationships among formal concepts allows us to describe not only concepts or groups of objects in isolation, but also a chain of concepts as they would appear as part of a navigation pathway as a user navigates across the concept lattice.

For neighbouring formal concepts, if a user moves to a formal concept's *subconcept* or *lower neighbour*, they navigate to a more specific formal concept. Likewise, if a user moves to a formal concept's *superconcept* or *upper neighbour*, they navigate to a more general formal concept. The *predicate map* – with its ability to prioritise which attributes are displayed for selected parts of a NL phrase – can be used to highlight concepts that are more general or more specific to one another, where attributes that describe more specific concepts (such

as the name of an artist) may be expressed in place of attributes that describe broader concepts (e.g. artist nationality). In Fig. 2 the display of the more specific artist attribute implies their nationality – the relevant domain knowledge either known to the user or specifically expressed once the user navigates to a broader superconcept, e.g. consider a formal concept with the attributes:

```
fromTimePeriod:20th century
artistNationality:American
fromTimePeriod:1920s
isTypeOf:photograph
byArtist:Consuelo Kanaga
```

CollectionWeb generates the NL statement:"photographs by Consuelo Kanaga from the 1920s". Note that the the attribute value of *artistNationality:American* is not displayed in the statement due to it being 'overridden' by the *byArtist:Consuelo Kanaga* attribute as shown in the predicate map in Fig. 2., and the *fromTimePeriod: 20th century* attribute is not shown because the *TimePeriodParser* omits century attributes when more specific date attributes are present. In this way, the predicate map can imply a hierarchy of attribute values that have different predicates, but share a common semantic class, e.g., given the *byArtist* attributes is more specific than *artistNationality* attribute, and the *designedIn* is more specific than *associatedWithLocation* , only the more specific *byArtist* and *designedIn* attributes are displayed – even when *artistNationality* and *associatedWithLocation* attributes are present.

If, following the same rules and predicate map as above, the attributes *fromTimePeriod:1920s* and *byArtist:Consuelo Kanaga* are 'removed' from the formal concept, i.e. by means of navigating to a formal concept's upper neighbour, then the NL statement would be: "photographs by American artists from the 20th century".

The expression of the broader terms – *American artists* and *20th century*, implies that the concept is broader. Hence, to a user, navigating between the two concepts appears as follows:

photographs by Consuelo Kanaga from the 1920s
⇓
more photographs by American artists from the 20th century

CollectionWeb pre-pends *more* to the concept the user is navigating to if an attribute with an *is* or *isTypeOf* predicate is present, or *other* if those attributes are not present, as shown in the following example where the user navigates to another upper neighbour where the *isTypeOf:photograph* is not present:

photographs by Consuelo Kanaga from the 1920s
⇓
more photographs by American artists from the 20th century
⇓
other works by American artists from the 20th century

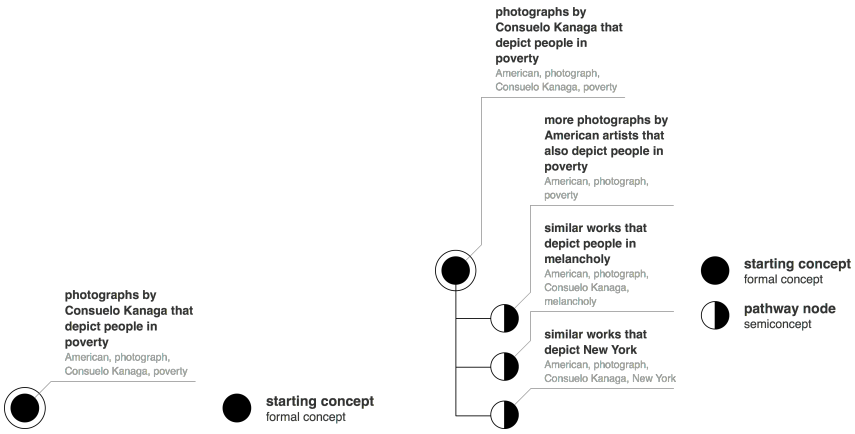

Fig. 3. (left) The concept pathway showing a list of its attributes {'American', 'photograph', 'Consuelo Kanaga', 'poverty'} and a generated NL expression "photographs by Consuelo Kanaga that depict people in poverty". (right) The start of a concept pathway with generated NL expressions, highlighting the similarities and differences between the starting concept and its child concepts.

If a user navigates between two concepts that do not have a subconcept/-superconcept relationship, CollectionWeb only displays the attributes in the concept that the user is navigating to that are not present in the concept that they are navigating from, e.g.,, consider the two attribute sets, each forming formal concepts that are similar:

```
artistNationality:American
isTypeOf:photograph
byArtist:Consuelo Kanaga
depicts:emotional-subject:poverty
```

```
artistNationality:American
isTypeOf:photograph
byArtist:Consuelo Kanaga
depicts:melancholy
depicts:person:mother
```

The new attributes introduced by the second formal concept *depicts: emotional-subject:melancholy* and *depicts:person:mother and child* are the only ones displayed when the user navigates to it, as shown in the example below:

photographs by Consuelo Kanaga that depict poverty
$\Downarrow$
similar photographs that depict mothers in melancholy

Note that attributes with *is* or *isTypeOf* predicates are always displayed. Continuing the example, if the user navigates to an upper neighbour of the second concept, removing the attribute *depicts: emotional-subject:melancholy*, its upper neighbour is displayed.

photographs by Consuelo Kanaga that depict poverty
⇓
similar photographs that depict mothers in melancholy
⇓
more works by Consuelo Kanaga that also depict mothers

These NL fragments can describe a user journey through a concept pathway. A concept pathway is a tree structure that forms as a user navigates between similar formal concepts.

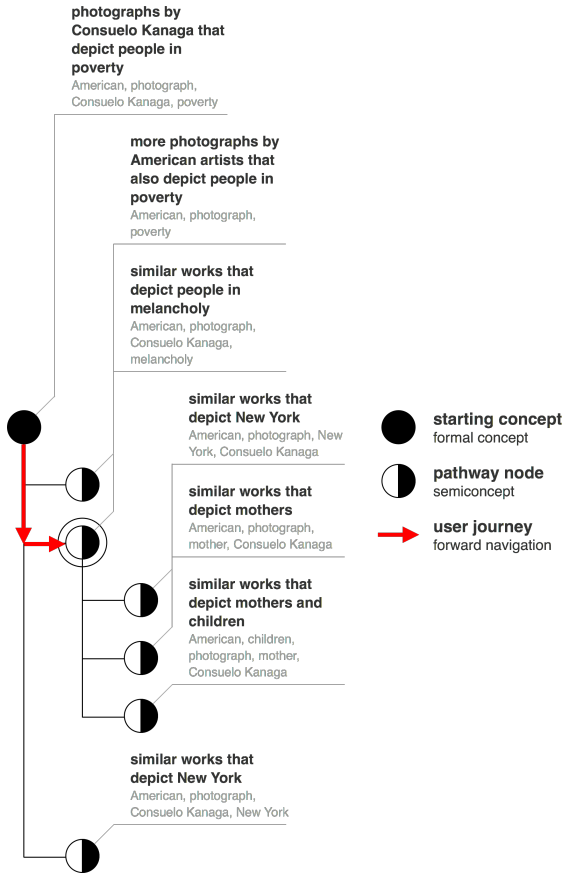

Fig. 4. The now expanded pathway structure showing a user navigating from "photographs by Consuelo Kanaga that depict people in poverty" to "similar works that depict people in melancholy", revealing more divergent concepts for navigation.

Since CollectionWeb expresses the difference between formal concepts that are broader or narrower, and highlights the differences between concepts that are similar, the framework can use its NL capabilities to describe a user's

experience through a pathway as a *narrative*. To demonstrate, the example that follows shows the concept-linked pathways with NL statements from the framework. The example begins with the starting concept, a selection of photographs by Consuelo Kanaga that depict people in poverty.

Fig. 3. shows semiconcepts, demonstrating that one could navigate to similar works that depict people in melancholy or works that depict the city of New York, or explore more photographs by American artists that depict people in poverty. These selections are based on the formal concepts that have been identified as 'most similar' to the photographs, given that they contain new objects that are not in the object set of the starting concept.

Fig. 4., shows navigating to the second formal concept, where the concept-linked pathway reveals semiconcepts that are thematically similar to "works that depict people in melancholy", but are further divergent from the original set of photographs by Consuelo Kanaga. Fig. 5. shows backtracking to the first pathway "photographs by Consuelo Kanaga that depict people in poverty" and selecting its third semiconcept – "similar works that depict New York" – and from that concept, its first child concept: "more photographs by American artists that also depict New York". The pathway navigation presented in Fig. 5. forms a narrative, shown in Fig. 5., that describe the user's navigation journey throughout the collection.

5 Conclusion

In this paper we have shown how an effective and simple ontology of predicates can be organised into a predicate map with accompanying parsers to generate NL descriptions of formal concepts. While enhancing usability of our case studies, the design goal of a more narrative like expression of formal concepts using natural language. The techniques described have been used in several museum-based system described in *Virtual Museum of the Pacific* [2], the *Brooklyn Museum Canvas* [7] and the *A Place for Art* [8] iPad app [4]. The innovation here is the application of natural language processing to formal concept analysis, in particular the systematic treatment of formal concepts as natural language fragments.

References

1. C. Carpineto and G. Romano. *Concept data analysis: Theory and applications*. J. Wiley, 2004.
2. P. Eklund, P. Goodall, and T. Wray. Cluster-based Navigation for a Virtual Museum. In *9th RIAO Conference – Adaptivity, Personalization and Fusion – of Heterogeneous Information*, Paris, April 2010. ACM Press.
3. J.R. Finkel, T. Grenager, and C. Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 363–370, 2005.

[4] https://itunes.apple.com/au/app/a-place-for-art/id638054832

4. A. Formica. Ontology-based concept similarity in Formal Concept Analysis. *Information Sciences*, 176(18):2624–2641, September 2006.
5. D. Roth and D. Zelenko. Part of Speech Tagging Using a Network of Linear Separators. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 1136–1142, 1998.
6. R. Wille and B. Ganter. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag, Berlin, 1999.
7. T. Wray and P. Eklund. Concepts and Collections: A Case Study using Objects from the Brooklyn Museum. In L Predoiu, S Hennicke, A Nurnberger, A Mitschick, and S Ross, editors, *Proceedings of the 1st International Workshop on Semantic Digital Archives*, pages 109–120, 2011.
8. T. Wray, P. Eklund, and K. Kautz. Pathways through Information Landscapes: Alternative Design Criteria for Digital Art Collections. In *ICIS 2013 Proceedings*, Milan, Italy, 2013.

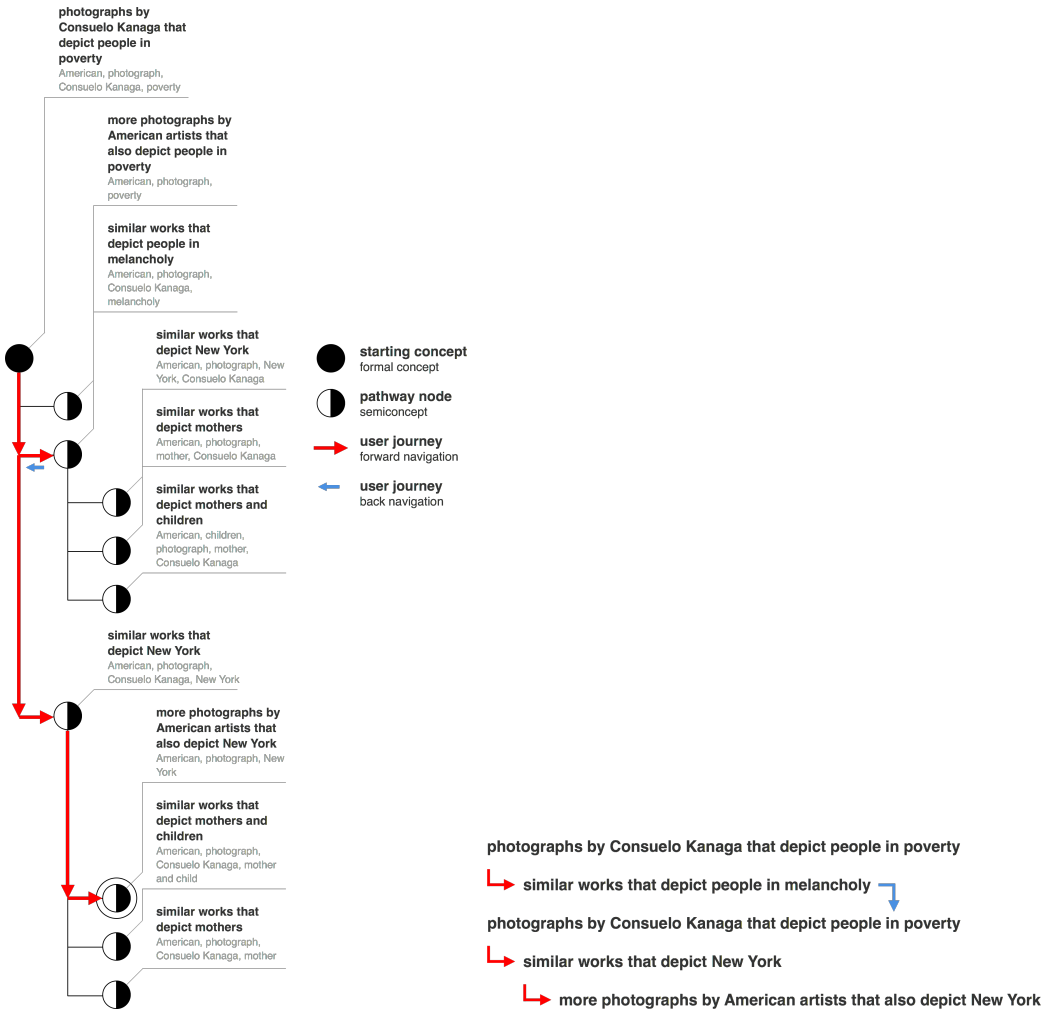

Fig. 5. (left) The expanded pathway structure depicting forward and backward navigations through a tree of thematically related concepts. (right) A narrative of pathway navigations from (left).